Automatic English-Chinese name transliteration for development of multilingual resources

Stephen Wan and Cornelia Maria Verspoor
Microsoft Research Institute
Macquarie University
Sydney NSW 2109, Australia
{swan, kversp}@mri.mq.edu.au

Abstract

In this paper, we describe issues in the translation of proper names from English to Chinese which we have faced in constructing a system for multilingual text generation supporting both languages. We introduce an algorithm for mapping from English names to Chinese characters based on (1) heuristics about relationships between English spelling and pronunciation, and (2) consistent relationships between English phonemes and Chinese characters.

1 Introduction

In the context of multilingual natural language processing systems which aim for coverage of both languages using a roman alphabet and languages using other alphabets, the development of lexical resources must include mechanisms for handling words which do not have standard translations. Words falling into this category are words which do not have any obvious semantic content, e.g. most indo-european personal and place names, and which can therefore not simply be mapped to translation equivalents.

In this paper, we examine the problem of generating Chinese characters which correspond to English personal and place names. Section 2 introduces the basic principles of English-Chinese transliteration, Section 3 identifies issues specific to the domain of name transliteration, and Section 4 introduces a rule-based algorithm for automatically performing the name transliteration. In Section 5 we present an example of the application of the algorithm, and in Section 6 we discuss extensions to improve the robustness of the algorithm.

Our need for automatic transliteration mechanisms stems from a multilingual text generation system which we are currently constructing, on the basis of an English-language database containing descriptive information about museum objects (the POWER system; Verspoor *et al* 1998). That database includes fields such as *manufacturer*, with values of personal and place names. Place names and personal names do not fall into a well-defined set, nor do they have semantic content which can be expressed in other languages through words equivalent in meaning. As more objects are added to our database (as will happen as a museum acquires new objects), new names will be introduced, and these must also be added to the lexica for each language in the system. We require an automatic procedure for achieving this, and concentrate here on techniques for the creation of a Chinese lexicon.

2 English-Chinese Transliteration

We use the term *transliteration* to refer generally to the problem of the identification of a specific textual form in an output language (in our case Chinese characters) which corresponds to a specific textual form in an input language (an English word or phrase). For words with semantic content, this process is essentially equivalent to the *translation* of individual words. So, the English word "black" is associated with a concept which is expressed as "黑" ([hēi]) in Chinese. In this case, a dictionary search establishes the input-output correspondence.

For words with little or no semantic content, such as personal and place names, dictionary lookup may suffice where standard translations exist, but in general it cannot be assumed that names will be included in the bilingual dictionary. In multilingual systems designed only for languages sharing the roman alphabet, such names pose no problem as they can simply be included unaltered in output texts in any of the languages. They cannot, however, be included in a Chinese text, as the roman characters cannot standardly be realized in the Han character set.

3 Name Transliteration

English-Chinese name transliteration occurs on the basis of pronunciation. That is, the written English word is mapped to the written Chinese character(s) via the spoken form associated with the word. The idealized process consists of:

1. mapping an English word (grapheme) to a phonemic representation
2. mapping each phoneme composing the word to a corresponding Chinese character

In practice, this process is not entirely straightforward. We outline several issues complicating the automation of this process below.

The written form of English is less than normalized. A particular English grapheme (letter or letter group) does not always correspond to a single phoneme (e.g. **ea** is pronounced differently in *eat*, *threat*, *heart*, etc.), and many English multi-letter combinations are realised as a single phoneme in pronunciation (so **f**, **ff**, **ph**, and **gh** can all map to /f/) (van den Bosch 1997). An important step in grapheme-phoneme conversion is the segmentation of words into syllables. However, this process is dependent on factors such as morphology. The syllabification of "hothead" divides the letter combination **th**, while the same combination corresponds to a single phoneme in "bother". Automatic identification of the phonemes in a word is therefore a difficult problem.

Many approaches exist in the literature to solving the grapheme-phoneme conversion problem. Divay and Vitale (1997) review several of these, and introduce a rule-based approach (with 1,500 rules for English) which achieved 94.9% accuracy on one corpus and 64.37% on another. Van den Bosch (1997) evaluates instance-based learning algorithms and a decision tree algorithm, finding that the best of these algorithms can achieve 96.9% accuracy.

Even when a reliable grapheme-to-phoneme conversion module can be constructed, the English-Chinese transliteration process is faced with the task of mapping phonemes in the source language to counterparts in the target language, difficult due to phonemic divergence between the two languages. English permits initial and final consonant clusters in syllables. Mandarin Chinese, in contrast, primarily has a consonant-vowel or consonant-vowel-[nasal consonant (/n/ or /ŋ/)] syllable structure. English consonant clusters, when pronounced within the Chinese phonemic system, must either be reduced to a single phoneme or converted to a consonant-vowel-consonant-vowel structure by inserting a vowel between the consonants in the cluster. In addition to these phonotactic constraints, the range of Chinese phonemes is not fully compatible with those of English. For instance, Mandarin does not use the phoneme /v/ and so that phoneme in English words is realized as either /w/ or /f/ in the Chinese counterpart.

We focus on the specific problem of country name transliteration from English into Chinese. The algorithm does not aim to specify general grapheme-phoneme conversion for English, but only for the subset of English words relevant to place name transliteration. This limited domain rarely exhibits complex morphology and thus a robust morphological module is not included. In addition, foreign language morphemes are treated superficially. Thus, the algorithm transliterates the *"-istan"* (a morpheme having meaning in Persian) of "*Afghanistan"* in spite of a standard transliteration which omits this morpheme.

The transliteration process is intended to be based purely on phonetic equivalency. On occasion, country names will have some additional meaning in English apart from the referential function, as in "*The United States*". Such names are often translated semantically rather than phonetically in Chinese. However, this in not uniformly true, for example "*Virgin*" in "*British Virgin Islands*" is transliterated. We therefore introduce a dictionary lookup step prior to commencing transliteration, to identify cases which have a standard translation.

The transliteration algorithm results in a string of Han characters, the ideographic script used for Chinese. While the dialects of Chinese share the same orthography, they do not share the same pronunciation. This algorithm is based on the Mandarin dialect.

Because automation of this algorithm is our primary goal, the transliteration starts with a written source and it is assumed that the orthography represents an assimilated pronunciation, even though English has borrowed many country names. This is permitted only because the mapping from English phonemes to Chinese phonemes loses a large degree of variance: English vowel monothongs are flattened into a fewer number Chinese monothongs. However, Chinese has a larger set of diphthongs and triphthongs. This results in approximating a prototypical vowel by the closest match within the set of Chinese vowels.

4 An Algorithm for Auto Transliteration

The algorithm begins with a *proper noun phrase* (PNP) and returns a transliteration in Chinese characters. The process involves five main stages: *Semantic Abstraction*, *Syllabification*, *Subsyllable Divisions*, *Mapping to Pinyin*, and *Mapping to Han Characters*.

4.1 Semantic Abstraction

The PNP may consist of one or more words. If it is longer than a single word, it is likely that some part of it may have an existing semantic translation. "*The*" and "*of*" are omitted by convention. To ensure that such words as "*United*" are tran-

slated and not transliterated[1], we pass the entire PNP into a dictionary in search of a standard translation. If a match is not immediately successful, we break the PNP into words and pass each word into the dictionary to check for a semantic translation[2]. This portion of the algorithm controls which words in the PNP are translated and which are transliterated.

```
Search for PNP in dictionary
    If exact match exists then
        return corresponding characters
    else
      remove article 'The' and preposition 'of'
      For each (remaining) word in PNP
        search for word in dictionary
        If exact match exists
            add matching characters to output string[3]
        else if the word is not already a chinese word
            transliterate the word and add to output string
```

4.2 Transliteration 1: Syllabification

Because Chinese characters are monosyllabic, each word to be transliterated must first be divided into syllables. The outcome is a list of syllables, each with at least one vowel part.

We distinguish between a consonant group and a consonant cluster, where a group is an arbitrary collection of consonant phonemes and a cluster is a known collection of consonants. Like Divay and Vitale (1997), we identify syllable boundaries on the basis of consonant clusters and vowels (ignoring morphological considerations). Any consonant group is divided into two parts, by identifying the final consonant cluster or lone consonant in that group and grouping that consonant (cluster) with the following vowel. The subsyllabification algorithm then further divides each identified syllable. While this procedure may not always strictly divide a word into standard syllables, it produces syllables of the form consonant-vowel, the common pronunciation of most Chinese characters.

4.2.1 Normalization

Prior to the syllabification process, the input string must be normalized, so that consonant clusters are reduced to a single phoneme represented by a single ASCII character (e.g. **ff** and **ph** are both reduced to **f**). Instances of 'y' as a vowel are also replaced by the vowel 'i'.

```
For each pair of identical consonants in the input string
    Reduce the pair to a singular instance of the consonant
For each substring in the input string listed in Appendix A
    Replace substring with the corresponding phoneme (App. A)
For all instances where 'y' is not followed by a vowel or 'y' follows a
consonant
    Replace this instance of 'y' with the vowel 'i'
When 'e' is followed by a consonant and an 'ia#'
    ;; (where # is the end of string marker)
    Replace the the preceding 'e' with 'i
```

4.2.2 Syllabification

```
If string begins with a consonant
    Then read/store consonants until next vowel and call this
    substring initial_consonant_group (or icg)
Read/store vowels until next consonant and call this substring
    vowels (or v)
If more characters, read/store consonants until next vowel and call
    this final_consonant_cluster (or fcc)
If length of fcc = 1 and fcc followed by substrings 'e#'
    final_vowel (or fv) = 'e'
    syllable = icg + v +fcc +fv
else if the last two letters of fcc form a substring in Appendix B
        then this string has a double consonant cluster
            next_syllable (or ns) = the last two letters of fcc
        reset fcc to be fcc with ns removed
    else
        next_syllable (or ns) = the last letter of fcc
    reset fcc to be fcc with ns removed
    syllable = icg + v + fcc
Store syllable in a list
Call syllabification procedure on substring [ns .. #]
```

4.3 Transliteration 2: Sub-syllable Divisions

The algorithm then proceeds to find patterns within each syllable of the list. The pattern matching consists of splitting those consonant clusters that cannot be pronounced within the Chinese phonemic set. These separated consonants are generally pronounced by inserting a context-dependent vowel. The Pinyin romanization consists of elements that can be described as consonants (including three consonant clusters "*zh*", "*ch*" and "*sh*") and vowels which consist of monothongs, diphthongs and vowels followed by a nasal /n/ or /ŋ/. Consonants that follow a set of vowels are examined to determine if they "modify" the vowel. Such consonants include the alveolar approximant /r/, the pharyngeal fricative /h/ or the above mentioned nasal consonants. These are then joined to the vowel to form the "vowel part". The "vowel part" may be divided so as to map onto a Pinyin syllable. Any remaining consonants are then split by inserting a vowel.

```
For each syllable s identified above
```

[1] The historical interactions of some European and Asian nations has lead to names that include some special meaning. Interaction with the dialects of the South may have produced transliterations based on regional pronunciations which are accepted as standard.

[2] There is some discrepency among speakers about the balance between translation and transliteration. For instance, the word 'New' is translated by some and transliterated by others.

[3] Identification of syntactic constraints is work-in-progress. Known nouns such as 'island' are moved to the end of the phrase while modifers (remaining words) maintain their relative order.

```
Initialize subsyllable_list (or sl) to the empty string
Identify initial_consonant_group sicg
While sicg is non-null
    If the first two letters of sicg appear in Appendix C
        then consonant_pair (or cp) = those two letters
            append cp to sl
            reset sicg to be the remainder of sicg
        else add the first letter of sicg to sl
            reset sicg to be the remainder of sicg
Identify vowels (v) in s
append v to last element of sl
identify final_consonant_cluster (fcc) of s
if sfcc is non-null
    if sfcc is equal to 'n', 'm', 'ng', 'h' or 'r'
        identify final vowels of s (sfv)
        If sfv exists and sfcc = 'n' or 'm'
            append sfcc to last element of sl
        else if sfv exists and sfcc not = 'n' or 'm'
            append sfcc + sfv to last element of sl
        else if sfv exists and sfcc = 'h' or 'r '
            discard sfcc + sfv
      else
        while sfcc is non null
            If the first two letters of sfcc appear in Appendix C
                then
                 cp = those two letters
                 append cp to sl
                 reset sfcc to be the remainder of sfcc
                else
                 add the first letter of sfcc to sl
                 reset sfcc to be the remainder of sfcc
For each element of sl
    If element does not include a vowel
        Insert context dependent vowel
```

This procedure will subdivide the syllable into pronounceable sections for mapping to the Chinese phoneme set. Thus each subsection should be of the form <cv>, <v> or <vc_n>, where "c" is a single consonant, "v" is a monothong or diphthong and "c_n" is a nasal consonant.

4.4 Transliteration 3: Mapping to Pinyin

The subsyllables are then mapped to the Pinyin romanization standard equivalents by means of a table (*Appendix D*). This table is indexed on the columns on the consonants of the subsyllable, and on the rows on the vowel part of the subsyllable. When an exact match cannot be found we prioritize aspects of the subsyllable. Often the highest priority is the initial consonant. Of next priority are nasal consonants. This may demand an alternate vowel choice if no such combination of phonemes exists in the table.

4.5 Transliteration 4: Mapping to Han

Once the Pinyin of a word is established, the Han characters are simply extracted from a table of specifying the Pinyin <cv> - Han character correspondence (*Appendix E*). In some cases, multiple characters might be possible but the table includes only the most common.

5 An Example

The transliteration of the place name "*Faeroe Islands*" according to the algorithm will proceed as follows:

1. No match for "*Faeroe*" in the dictionary, so must be transliterated :
2. Divide Faeroe into two syllables by recognizing the syllabic break falls before the "*r*" in the middle consonant group.
3. Map /fae/ and /roe/ onto their Chinese equivalents. Since no vowel form /ae/ exists in Chinese, this is mapped to /ei/. The /r/ of the second syllable is mapped to /l/ and /oe/ is correspondingly mapped to /uo/.
4. Since each syllable is of the form <cv>, no subsyllabic processing is required.
5. The transliterated phrase "fei luo" is the mapped to the Han characters: "非罗"
6. "Islands" is searched for and found in the dictionary : "群岛" (qún d o)
7. The characters of the translated "*Islands*" are placed after the transliteration of "*Faeroe*" : "非罗群岛" *(f i l o qún d o)*

6 Conclusions and Future Extensions

The algorithm we have outlined is being implemented as a tool for the creation of Chinese lexical resources within a multilingual text generation project from an English-language source database. We focused on the requirements of the domain of English place names. The algorithm is currently being extended to include personal name transliteration as well, which requires a different set of characters. A personal name transliteration standard has been developed and is in use in China (Chanzhong Wu, p.c.). By mapping the Pinyin transliterations arrived by our algorithm to this different set of characters, we can extend the domain to include personal names.

In its present form, the algorithm will not always generate transliterations matching those which might be produced by a human transliterator due to the influence of historical factors or individual differences. However, the aim of the algorithm is to produce a transliteration understandable by readers of a Chinese text. While the algorithm mimics the intuitive superimposition of phonemic and phonotactic systems, the ultimate goals of the algorithm are generality and reliability. Indeed, the result from the example above corresponds to a standard transliteration. Thus the algorithm produces results which are recognisable. The degree to which the transliteration is recognised by the human speaker is dependent in part on the length of the original name. Longer names with many syllables are less recognisable than shorter names. The introduced phonemic conversion rules are merely those most common

and further work will strengthen the generality of the tool. Further research will include a more formal analysis of the correspondences between English and Chinese phonemes. Furthermore, the algorithm is far from robust due to its current limited focus, and errors made in earlier stages are propagated and possibly magnified as the algorithm continues. Since place names and people's names originate from many cultures, this algorithm will not produce desirable results unless the written form exhibits some assimilation to English spelling. We are currently investigating the application of lazy learning techniques (as described by van den Bosch 1997) to learning the English naming word-phoneme correspondences from a corpus of names. Such a module could eventually replace our simplistic rule-based procedure, and could feed into the phoneme-Pinyin mapping module, ultimately resulting in greater accuracy.

The applications of such an algorithm are countless. Currently, the process of finding a less common country, city, or county name is an arduous procedure. Because transliteration uses no semantic content, it is a obvious task for automation. This algorithm could also be applied in the character entry on a Chinese word processor or to index Chinese electronic atlases. When attached to a robust grapheme-to-phoneme module, the transliteration into Chinese characters is ultimately a mapping to Chinese-specific IPA phonetics, raising the possibility of speech synthesis of English names in Chinese, given that Pinyin is a phonemically normalized orthography.

Acknowledgements

Our thanks go to Canzhong Wu for help with identifying Chinese mappings, and the members of Dynamic Document Delivery project at the Microsoft Research Institute (the POWER team).

References

Divay M. and Vitale A.J. (1997*) Algorithms for Grapheme-Phoneme Translation for English and French: Applications*. Computational Linguistics, 23/4, pp. 495—524.

Verspoor, C., Dale, R., Green, S., Milosavljevic, M., Paris, C., and Williams, S. (1998) *Intelligent Agents for Information Presentation: Dynamic Description of Knowledge Base Objects*. In the proceedings of the International Workshop on Intelligent Agents on the Internet and Web, Mexico City, Mexico, 16-20 March 1998, pp. 75-86.

van den Bosch A. (1997) *Learning to pronounce written words: A study in inductive language learning*. PhD thesis, University of Maastricht, Uitgeverij Phidippides, Cadier en Keer, the Netherlands, 229p.

Appendices A. B. and C. English-Chinese unitary consonant correspondences, consonant pairs, and double consonant correspondences

bh => b	cqu => k
ngh => ngh	sc => c
gh => gh	dj => j
ph => f	ts => c
th => t	lk => k
ck => k	we => w
r + cons. => cons.	

tr	bl
sh	cl
ch	fl
cz	kl
sp	pl
st	sl
sw	

cz => ch	sp => xi b-
st => shi d-	sw => ru-
ch => ch	sh => sh

Appendix D. Portion of English phoneme – Chinese Pinyin Mapping Table

	f-	n-	p-	r-	v
a	fa	na	ba	la	wa
ae	fei	nei	bei	lei	wei
ai	fei	nei	bei	lei	wei
ai	fai	nai	bai	lai	wai
ai	fa yi	na yi	ba yi	la yi	wa yi
ao	-	nao	bao	lao	-
ar#	-	nuo	-	luo	wuo
au	-	nuo	-	luo	wuo
ay	fei	nei	bei	lei	wei
o	fo	-	bo	-	wo
o#	-	nuo#	-	luo#	wuo#
oa	-	-	bo ya	-	wo ya
oe	-	nuo	-	luo	wuo
oi	-	-	-	-	#
on	-	-	-	lun	-
or#	-	nuo#	-	luo#	wuo#
ou	-	nuo	-	luo	wuo

Appendix E. Pinyin-Han table (portion)

a;阿	di;迪	hong;洪	lun;伦	qi;其	wang;旺
ai;埃	dian;典	ji;几	luo;洛	qiu;求	wang;汪
ai;爱	dian;甸	ji;及	luo;罗	ri;日	wei;伟
an;安	du;度	ji;吉	luu;律	rui;士	wei;危
an;岸	du;都	ji;基	ma;马	rui;瑞	wei;委
ang;昂	dun;敦	ji;济	mai;买	sa;萨	wei;威
ao;澳	duo;多	jia;加	mai;麦	sai;塞	wei;维
ba;巴	e;俄	jian;柬	man;曼	sang;桑	wen;文
bai;百	e;厄	jie;捷	mao;毛	se;色	wu;乌
ban;班	er;尔	jin;津	mei;美	sen;森	wuo;挝
bao;保	er;耳	jing;京	men;门	sha;沙	xi;希
bao;堡	fa;伐	ju;克	meng;古	shao;绍	xi;西
bei;北	fei;斐	ka;卡	meng;蒙	she;舌	xian;鲜
bei;贝	fei;菲	ka;喀	meng;孟	shi;士	xiang;象
ben;本	fei;非	kai;开	mi;密	shi;市	xiang;香
bi;比	fen;芬	ke;库	mi;秘	shi;时	xin;新
bing;冰	fo;佛	ke;科	mi;米	shi;瑞	xiong;匈
bing;宾	fu;夫	ken;肯	mian;缅	si;斯	xu;叙
bo;伯	fu;富	la;拉	mo;墨	song;松	ya;亚
bo;博	fu;福	la;腊	mo;摩	su;苏	ya;牙
bo;泊	gan;干	lai;来	mo;莫	suo;所	ye;也
bo;波	gang;冈	lan;兰	mu;慕	suo;索	yi;以
bo;玻	gang;刚	lang;朗	na;拿	ta;他	yi;伊
bu;不	gang;港	lao;老	na;纳	ta;塔	yi;意
bu;布	ge;各	le;勒	na;那	tai;台	yin;印
bu;浦	ge;哥	li;列	nan;南	tai;汰	yue;约
chao;朝	ge;格	li;利	nao;瑙	tai;泰	yue;越